

28, Ln. 32-41), a Link List (Dynamic Link 220d, Col. 29, Ln. 62-66), a Stack Frame Extension (Shadow Stack 212, Col. 30, Ln. 3-6). Tye is silent with reference to run0time environment, a current thread and a thread identifier. Miller teaches a Run-Time Environment (Run-Time Environment 122), a Current Thread (Thread 204), a Thread Identifier (Thread 204).

It would have been obvious to apply the teaching of Miller to the system of Tye. One would have been motivated to make such a modification so that the method and current thread can be used to access the shadow stack.

As to claim 19, see the rejection of claim 1.

As to claim 26, see the rejection of claim 1.

Claim 10, which is representative of claims 19 and 26 with regard to similarly recited subject matter, reads as follows:

10. A process for providing a privilege for a method of a current thread that is currently executing in a run-time environment in a data processing system, the run-time environment having a stack comprising stack frames with stack frame pointers for associated methods, the process comprising the computer-implemented steps of:

using a thread identifier of the current thread, locating a linked list;

and

searching the linked list for an entry having a stack frame pointer that matches the stack frame pointer of the method, wherein an entry of the linked list is a stack frame extension. (emphasis added)

Tye teaches a method and apparatus for data flow analysis in which a computer system provides a native instruction or native instruction routine when presented with a non-native image of an application program written for a non-native instruction set. A run-time system collects profile data in response to execution of native instructions to determine execution characteristics of non-native instructions. The profile statistics and the non-native instructions are fed to a binary translator which generates a translated native image of the non-native instructions.

Miller teaches a system and method for managing control flow of computer programs executing in a computer system. The system of Miller manages an invocation stack which includes a plurality of stack frames. A stack frame is selected and a

determination is made as to whether a user specified event processing procedure has been registered with the selected stack frame which is capable of handling a current event. If so, the event is processed using the specified event processing procedure.

While Tye generally illustrates a method, a computing system, a stack, stack frames, and a linked list, Tye has nothing to do with the specific features of the present invention as recited in claim 10. Specifically, Tye is not directed to a process for providing a privilege for a method of a current thread. To the contrary, Tye is directed to a method for data flow analysis for generating native images of non-native instructions. Tye does not even mention privileges, let alone providing privileges for a method in a current thread. The Office Action alleges that Tye teaches privileges as the "Call to B" or "Call to B'" in Figure 23, however these elements 233 and 243 of figure 3 are merely instructions (see column 29, lines 29-30 and lines 40-44). They are not privileges.

Likewise, Miller has nothing to do with the invention recited in claim 10. Miller, like Tye, does not mention privileges or providing privileges for a method in a current thread. The Office Action alleges that Miller teaches a run-time environment, a thread and a thread identifier. While Miller may teach a run-time environment and a thread, there is nothing in Miller that teaches or suggests a thread identifier, or the specific manner by which a thread identifier is used as recited in the claim 10.

Neither Tye nor Miller teach or suggest locating a linked list using a thread identifier of a current thread or searching a linked list for an entry having a stack frame pointer that matches a stack frame pointer of a method, wherein an entry of the linked list is a stack frame extension, as recited in claim 10. The Office Action does not even address these features. Rather, the Office Action merely provides a laundry list of elements that are allegedly taught in the Tye and Miller references and somehow comes to the conclusion that the invention recited in claim 10 is obvious. Applicants respectfully submit that, while some of the parts, e.g., a stack, stack frames and a linked list, are generally taught in the references, the assembly of parts as explicitly recited in claim 10 is not taught or suggested by Tye or Miller, either alone or in combination. Where the claims require that an element serve a specific purpose, the fact that a similar element was used for another purpose in the prior art or that the claimed element has a prior

art attribute does not establish a *prima facie* case of obviousness. *In re Wright*, 848 F.2d 1216, 6 U.S.P.Q.2d 1959 (Fed. Cir. 1988).

Where in Tye or Miller is it taught to locate a linked list using a thread identifier of a current thread? Neither Tye or Miller teach or suggest locating a linked list, let alone locating a linked list using a thread identifier of a current thread. While Figure 22 of Tye illustrates the shadow stack 212 as a linked list (the *dylnk* 220d points to a previous shadow frame header 214), there is no teaching or suggestion in Tye to locate the shadow stack 212 using a thread identifier of a current thread. The mere general teaching of a linked list does not teach or suggest to one of ordinary skill in the art to locate a linked list using a thread identifier of a current thread. In fact, the Office Action admits that Tye does not teach or suggest a current thread or a thread identifier, therefore it is impossible to locate a linked list using a thread identifier of a current thread in the system of Tye.

Where in Tye or Miller is it taught to search a linked list for an entry having a stack frame pointer that matches a stack frame pointer of a method? Although Tye mentions a non-native stack pointer 220a and a shadow stack pointer 221, there is no teaching or suggestion to search a linked list for an entry having a stack frame pointer that matches a stack frame pointer of a method. There is no determination made in Tye as to whether or not the non-native stack pointer 220a or the shadow stack pointer 221 matches a stack frame pointer of a method. Rather, the shadow stack 212, and hence the shadow stack pointer 221, is generated when routine code that has been translated is being executed in a native architecture. As a result, translated code uses the shadow stack 212 while the interpreted code uses the non-native stack 211. While the shadow stack 211 includes a field 220b in which a non-native return address is stored, there is no search of the shadow stack 211 for an entry having a stack frame pointer that matches a stack frame pointer of a method.

Since the Office Action merely alleges that the references teach various parts recited in claim 10 and does not address the specific manner by which these parts are utilized, as recited in claim 10, the Office Action has not established a *prima facie* case of obviousness. Claim 10 of the present application does not claim merely a data processing system a stack, stack frames, a stack frame pointer, a stack frame extension, a run-time environment, a current thread, and a thread identifier, as the Office Action

alleges. Rather, claim 10 recites using a thread identifier of the current thread to locate a linked list and searching the linked list for an entry having a stack frame pointer that matches the stack frame pointer of a method. None of these features are present in either Tye or Miller.

In view of the above, Applicants respectfully submit that neither Tye nor Miller, either alone or in combination, teach or suggest the features of independent claims 10, 19 and 26. At least by virtue of their dependency on claims 10 and 19, respectively, neither Tye nor Miller teach or suggest the features of dependent claims 11-16 and 20-25. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 10-16 and 20-25 under 35 U.S.C. § 103(a).

In addition to the above, neither Tye nor Miller teach or suggest the specific features recited in dependent claims 11-16 and 20-25. For example, with regard to claims 11 and 20, neither Tye nor Miller teach or suggest locating a linked list within a stack frame shadow apparatus comprising a plurality of linked lists, each linked list of the plurality of linked lists being associated with a thread. The Office Action alleges that simply because Tye teaches a shadow stack and a dynamic link that somehow this is the same as locating a linked list within a stack frame shadow apparatus comprising a plurality of linked lists. First, the Tye reference does not teach a stack frame shadow apparatus in which a plurality of linked lists exists. Rather, the shadow stack 212 contains a single linked list.

Second, there is no teaching or suggestion in either Tye or Miller, to locate a linked list within a stack frame shadow apparatus that comprises a plurality of linked lists. There is not teaching or suggestion to perform such locating because the stack shadow only contains a single linked list. Furthermore, the Office Action does not address this specific feature but again, only erroneously alleges that a stack frame shadow apparatus and a plurality of link lists are taught. The Office Action does not mention where the teaching or suggestion may be found in either reference to locate a linked list in a stack frame shadow apparatus that comprises a plurality of linked lists.

With regard to claims 12 and 21, neither Tye nor Miller teach or suggest that a stack frame extension comprises a stack frame pointer of a method, privilege information, and validation information. The Office Action alleges that these features are the same as

the stack pointer SP 220a, the non-native instruction pointer value IP 220b, and the native return address RET 220c of Tye. None of these elements in Tye are privilege information. How does a non-native instruction pointer value act as a privilege? There is no teaching or suggestion in Tye that non-native instruction pointer values operate as privilege information.

Regarding claims 14 and 23, neither Tye nor Miller teach or suggest adding an entry to a linked list if no matching entries are found in response to a request to enable a privilege for a method. The Office Action alleges that these features are taught by Tye because Tye teaches that shadow frame 214 is pushed onto the shadow stack 212 in column 28, lines 65-66 and column 29, lines 1-7. The pushing of a shadow frame 214 onto the shadow stack 212 is done in response to an instruction invoking a routine that has been translated, such as routine B' (see column 29, lines 40-49). The pushing of a shadow frame onto the shadow stack is not in response to a request to enable a privilege and is not performed if no matching entries are found in response to a request to enable a privilege for a method.

With regard to claims 15 and 24, neither Tye nor Miller teach or suggest removing a matching entry from a linked list if a matching entry is found in response to a request to revert a privilege for a method. The Office Action alleges that these features are taught in Tye at column 30, lines 3-40 and column 31, lines 1-28. While these sections mention reverting back to an interpreter if the non-native code is not well behaved with respect to depth in the non-native return address stack and the code is not well behaved with respect to the return address (see column 30, lines 14-19), there is nothing in these sections of Tye, or any other section of Tye, that teaches removing a matching entry from a linked list if a matching entry is found in response to a request to revert a privilege for a method. In fact, this section of Tye does not even mention removing anything from the shadow stack.

Regarding claims 16 and 25, neither Tye nor Miller teach or suggest retrieving privilege information and validation information for a matching entry from a linked list if a matching entry is found in response to a request to retrieve privileges for a method. The Office Action alleges that these features are taught in the same sections of Tye as set forth above with regard to claims 15 and 24, i.e. column 30, lines 3-4 and column 31,

lines 1-28. These sections make no mention what-so-ever of privileges or validation information, let alone retrieving privilege and validation information for a matching entry from a linked list if a matching entry is found in response to a request to retrieve privileges for a method. There is not teaching of identifying a matching entry in a linked list; there is no teaching of retrieving privilege or validation information; and there is no teaching of a request to retrieve privileges for a method. Thus, there cannot be any teaching in Tye of retrieving privilege and validation information for a matching entry from a linked list if a matching entry is found in response to a request to retrieve privileges for a method.

II. 35 U.S.C. § 103, Alleged Obviousness of Claim 17

Regarding independent claim 17, the Office Action states:

As to claim 17, claim 1 meets claim 17 expect for storing privilege information, querying a stack frame shadow apparatus, deleting privilege information in a stack frame shadow apparatus to revert a privilege for a method and a method.

Tye teaches Storing privilege information (“...Shadow Frame is allocated...”, Col. 29, Ln. 44-54), Querying a stack frame shadow apparatus (“The shadow instruction call B’ causes the shadow stack to be provided...”, Col. 29, Ln. 45-54) and Deleting privilege information in a stack frame shadow apparatus to revert a privilege for a method (“...revert...”, Col. 29, Ln. 14-19). Tye is silent with reference to a run-time environment.

Miller teaches a Run-Time Environment (Run-Time Environment 122). It would have been obvious to apply the teaching of miller to system of Tye. One would have been motivated to make such a modification in order utilize the efficiency of a run-time system.

Claim 17 recites:

17. A process for enabling and reverting a privilege for a method of a current thread that is currently executing in a run-time environment in a data processing system, the run-time environment having a stack comprising stack frames with stack frame pointers for associated methods, the process comprising the computer-implemented steps of:

storing privilege information in a stack frame shadow apparatus to enable a privilege for a method;

querying a stack frame shadow apparatus for privilege information for a method; and
deleting privilege information in a stack frame shadow apparatus to revert a privilege for a method. (emphasis added)

There is no teaching or suggestion in either Tye or Miller to store privilege information in a stack frame shadow apparatus, query a stack frame shadow apparatus for privilege information for a method, or revert a privilege of a method by deleting privilege information in a stack frame shadow apparatus. The Office Action alleges that Tye teaches storing privilege information in a shadow apparatus at column 29, lines 44-54. This section of Tye reads as follows:

The shadow frame is allocated at the beginning of a routing for all calls that the routine can make. The instruction Call B' causes the shadow stack to be provided with a shadow stack frame 14 which comprises the four above-mentioned fields 20a-20d and the optional fields for local storage. Thus, in field 20a is provided the contents An of the stack pointer (SP) 17 of the non-native return stack 11. This value corresponds to the location where the return address stored in the non-native return address stack 211 for the corresponding native instruction execution will be found.

Where in here is there any mention of privileges or storing privileges in a stack frame shadow apparatus. While this section of Tye mentions a shadow frame and a shadow stack, there is no teaching or suggestion to store privilege information in the shadow stack. To the contrary, the only information stored in the shadow stack is in the fields 220a-220d which include a stack pointer 220a, a non-native return address IP 220b, a native return address 220c and a dynamic link dynlk 220d to other entries in the shadow stack (see Figure 22). There is no privilege information stored in the shadow stack of Tye.

Furthermore, no other section of Tye teaches or suggests anything remotely similar to the features noted above in claim 17. Thus, despite the allegations of the Office Action, in actuality, Tye does not teach or suggest storing privilege information in a stack frame shadow apparatus.

Moreover, since the shadow stack of Tye does not store privilege information, it is impossible to query the shadow stack to obtain privilege information. Furthermore, it

is impossible to delete privilege information from the shadow stack of Tye because the shadow stack in Tye does not store privilege information. Thus, the Office Action is in error when it alleges that all of these features are taught by Tye.

Miller does not provide for the deficiencies of Tye noted above. That is, Miller does not teach or suggest storing privilege information in a stack frame shadow apparatus, querying a stack frame shadow apparatus for privilege information for a method, or reverting a privilege of a method by deleting privilege information in a stack frame shadow apparatus. Furthermore, the Office Action does not show where any of these features may be found in Miller.

Therefore, neither Tye nor Miller, either alone or in combination, teach or suggest the features of independent claim 17. Accordingly, Applicants respectfully request withdrawal of the rejection of claim 17 under 35 U.S.C. § 103(a).

III. 35 U.S.C. § 103, Alleged Obviousness of Claim 18

Regarding independent claim 18, the Office Action states:

As to claim 18, Tye teaches a Data Structure (shadow Stack 212), a Computer-readable medium (Main Memory 14, Disk 15, Disk 17), a Data Processing System (Computer System 10), a Set of stack frame extensions (Shadow Stack 212), a Pointer (SP 220a, dylnk 220d), a Stack Frame (Frame 214), a Method (Routine A, Routine B), a Data Field (SP 220a, IP 220b, RET 220c, dylnk 220d), and a Link List (dylnk 220d). Tye is silent with reference to a thread identifier.

Miller teaches a thread identifier (Thread 204). It would have been obvious to apply the teaching of Miller to the system of Tye. One would have been motivated to make such a modification in view of the fact that every thread has an associated stack frame.

Claim 18 recites:

18. A data structure on a computer-readable medium for use in a data processing system, the data structure comprising:
a set of stack frame extensions, wherein a stack frame extension comprises:

a pointer to a stack frame for a method;
a data field for privilege data for the method;
a data field for validation data for the method; and

a linked list of stack frame extension entries, wherein the linked list is identifiable by a thread identifier.

Neither Tye nor Miller teach or suggest the features of a data field for privilege data for a method or a data field for validation data for a method, as recited in claim 18. It is noted that the Office Action does not even address these features but merely states that Tye teaches a data field. Where in Tye is the data field that stores privilege data? Where in Tye is the data field that stores validation data? The data fields that the Office Action refers to are again 220a-220d which have been addressed above in the rejection of claim 17. None of these fields store privilege data or validation data.

Miller does not provide for the deficiencies of Tye. Miller does not teach a data field for privilege information or a data field for validation information. Miller is only cited as allegedly teaching a thread identifier, which in fact it does not teach as discussed above with regard to independent claim 10. There is no teaching in Miller of any of the features in claim 18 noted above.

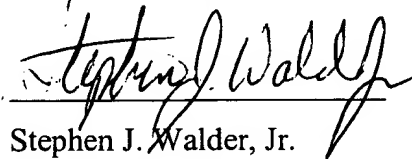
Thus, neither Tye nor Miller, either alone or in combination, teach or suggest the features of claim 18. Accordingly, Applicants respectfully request withdrawal of the rejection of claim 18 under 35 U.S.C. § 103(a).

IV. Conclusion

It is respectfully urged that the subject application is patentable over Tye and Miller and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: August 1, 2002

Respectfully submitted,



Stephen J. Walder, Jr.

Reg. No. 41,534

Carstens, Yee & Cahoon, LLP

P.O. Box 802334

Dallas, TX 75380

(972) 367-2001

Attorney for Applicants